CEWES MSRC/PET TR/99-23

Parallelization of a Coupled Hydraulics and Sediment Transport Model

by

P. V. Bangalore
J. Zhu
D. Huddleston
A. Skjellum
D. J. S. Welsh
K. W. Bedford
R. Wang
P. Sadayappan

Work funded by the DoD High Performance Computing Modernization Program CEWES Major Shared Resource Center through

Programming Environment and Training (PET)

Supported by Contract Number: DAHC94-96-C0002
Nichols Research Corporation

Parallelization of a Coupled Hydraulics and Sediment Transport Model

P. V. Bangalore, J. Zhu, D. Huddleston, A. Skjellum
Engineering Research Center for
Computational Field Simulation
Mississippi State University, Mississippi State, MS 39762

D. J. S. Welsh and K. W. Bedford
Department of Civil and Environmental Engineering and Engineering Graphics
The Ohio State University
Columbus, OH 43210

R. Wang and P. Sadayappan
Department of Computer and Information Science
The Ohio State University
Columbus, OH 43210

Abstract

The development of a fully coupled and parallel hydraulics, sediment, and wave model for simulation of physical marine processes has impact on many national defense and security operations. This report describes the development of a parallel version of a combined hydraulics and sediment model - a key component for the fully coupled model. The parallel version of the coupled hydraulics and sediment model was developed from the previously documented sequential version of the curvilinear hydrodynamics code in three dimensions with non-cohesive sediment transport (CH3D-SED). The primary purpose of this report is to document the development and verification of the parallel version of the coupled code. The parallel code was tested for two test data sets on the CRAY T3E.

1 Introduction

The three-dimensional Curvilinear Hydrodynamics model (CH3D-WES) developed at the U.S. Army Engineer Waterways Experiment Station provides a suitable framework for the simulation of unsteady, three-dimensional fixed-bed flow along deep navigational channels and irregular shorelines [1]. This hydrodynamic model was extended to include mobile-bed processes as described in [4]. The combined hydraulics and sediment models (CH3D-SED) provide a computational framework to study and model several aspects of sediment-flow interaction. The combined model has been used in the successful simulation of sedimentation on bendways, crossings, and distributaries on the lower Mississippi and Atchafalaya Rivers for the purpose of evaluating processes like dredging, channel evolution, and channel training processes [4]. Hence the CH3D-SED model was chosen in this project coupling the hydraulics, sediment and wave models [6]. The development of a fully coupled and scalable model for simulation of physical marine processes has impact on many national defense and security operations. Some of the important activities include:

- harbor access for Naval operations,
- wind/wave hazard forecast for fleet operations,
- coastal condition forecasts for amphibious landing activities,
- predicting ocean sediment storms in North Atlantic for finding submarines,
- integrated system for calculating subsurface mine burial mechanics and weapons retrieval,
- improved predictions of acoustic wave guides and channels for Naval submarine operations.

In order to have an effective coupling of the hydraulics, sediment, and wave models a scalable version of the coupled hydraulics and sediment models forms an important component. Initial development of a scalable version of the hydraulics model was described in [7]. Results herein describe the development of the scalable version of the coupled hydraulics and sediment models as an extension of the parallelization of the hydraulics model. In section 2 a brief description of the hydraulics model is presented and in section 3 the sediment model implemented within the hydraulics model is described along with the interactions between the flow and sediment computations. The details on the development of the parallel version of the coupled codes along with the parallelization strategy is presented in section 4. Section 5 provides details about the test cases and the performance results for the parallel version of the code on the CEWES MSRC platforms. Summary and conclusions are provided in section 6 and the last section outlines some of the future enhancements that can be performed.

2 Hydraulics Model

The three-dimensional hydrodynamic model (CH3D-WES) was developed at the U.S. Army Engineer Waterways Experiment Station as described in [1] to study flow processes impacting circulation and vertical mixing along irregular shorelines and deep navigation channels. The model includes physical processes such as tides, wind, salinity and temperature effects, freshwater inflows, turbulence effects, and the effect of the earth's rotation. The hydraulics model uses a boundary-fitted grid in the horizontal plane which provides detailed resolution to represent complex horizontal geometries found in deep navigation channels and irregular shorelines. The model uses the same number of vertical layers at each point on the horizontal grid, but the thickness of the layers scales with local depth.

Finite differences are used to replace the derivatives in the governing partial differential equations for an incompressible fluid, resulting in a system of linear algebraic equations to be solved in both the external and internal modes. In the external mode, the vertically averaged equations are solved to obtain a solution for the free surface displacement (S) and vertically averaged velocities ($\bar{U}$ and $\bar{V}$). In the transformed vertically averaged continuity equation all terms are treated implicitly, whereas, in the transformed vertically averaged momentum equations only the water surface slope terms are treated implicitly. The resulting finite difference equations are then solved using an ADI scheme such that the solution for the new time-step is obtained by first sweeping in the ξ-direction and then in the η-direction.

In the internal mode, the three velocity components ($\bar{u}$, $\bar{v}$, and w), salinity, and temperature are computed. The horizontal velocity components of the 3D velocity are calculated by first computing the deviation of the horizontal components of the full 3D velocity from the vertically averaged velocity ($\bar{u'}$ and $\bar{v'}$) and then adding it to the vertically averaged velocity components, that is, $\bar{u} = \frac{\bar{U}}{H} + \bar{u'}$ and $\bar{v} = \frac{\bar{V}}{H} + \bar{v'}$. To ensure that the vertical integration of the $(\bar{u'}, \bar{v'})$ is zero, the nonlinear inertia and turbulent diffusion terms in the vertically averaged momentum equations are evaluated by summing the corresponding terms in the 3D equations at all vertical layers. The vertical diffusion terms are treated implicitly in all equations and the bottom friction in the momentum equations is treated implicitly. The convective terms in the momentum equations are represented using Roache's second order upwind differencing and the advective terms in the transport equations for the salinity and temperature are represented using a spatially and temporally third-order scheme called QUICKEST [2]. A vertical k-ϵ turbulence eddy viscosity model is used to model the wind shear, bottom shear, velocity gradient turbulence production, dissipation, diffusion and stratification.

The computational domain uses a staggered grid in both the horizontal and vertical directions. Horizontal velocities, temperature, salinity, density and turbulence quantities are computed at cell centers (or "half" grid points) whereas vertical velocities are computed at cell faces (or "full" grid points). The two arrays NS and MS are used to denote the boundary types for each cell. The NS array denotes the left and right boundary types whereas the MS array denotes the top and bottom boundary types. Figure 1 describes the main computational loop performed by the hydraulics model.

3 Sediment Model

A non-cohesive sediment model was added to the hydraulics model to include mobile bed processes (such as bedload transport, suspended-sediment transport, bed level changes, hydraulic sorting and armoring, etc.) as described in [4]. The model is based on the concept of an active layer at the top of the bed in which erosion, deposition, and sediment transport occur. Further, improvements were made to the sediment modeling of the hydraulics model to reduce the time taken for sediment computations as described in [5], and this sequential version of the model is used as the starting point of the current work. The important aspects of mobile bed process that are considered by the sediment model are [4]:

- suspended-sediment transport, bedload transport, and interaction between the two

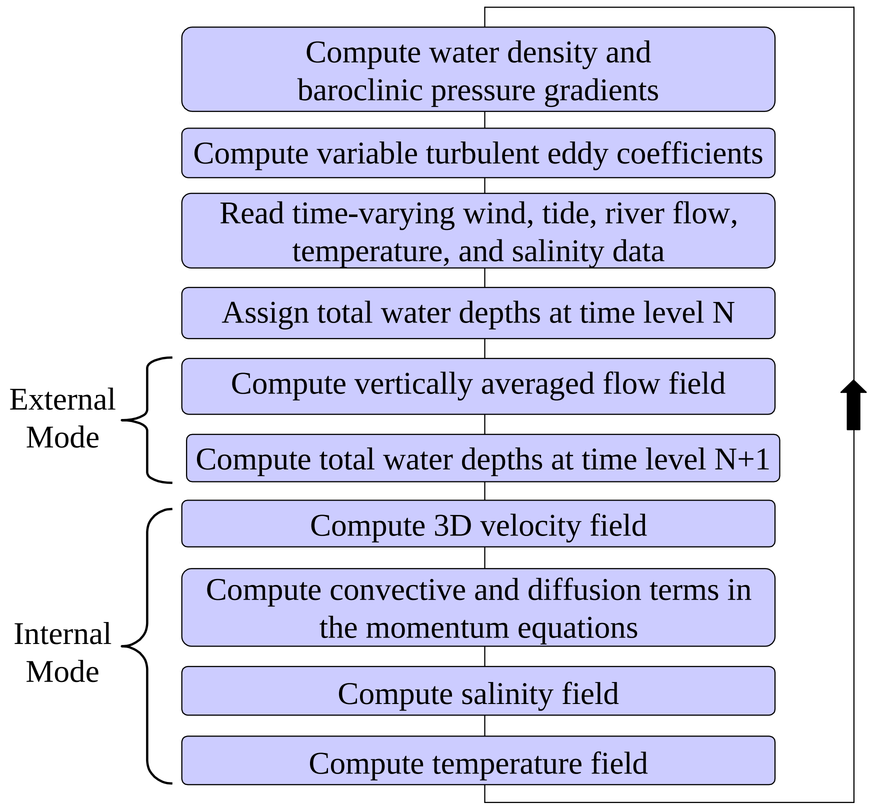
Compute water density and baroclinic pressure gradients
Compute variable turbulent eddy coefficients
Read time-varying wind, tide, river flow, temperature, and salinity data
Assign total water depths at time level N
External Mode
Compute vertically averaged flow field
Compute total water depths at time level N+1
Internal Mode
Compute 3D velocity field
Compute convective and diffusion terms in the momentum equations
Compute salinity field
Compute temperature field

Figure 1: Main computational loop in the Hydraulics Model

- bed level changes
- differential settling
- hydraulic sorting and armoring
- interaction between the changes in bed elevation and bed surface size distribution with flow field
- washload transport

The sediment computations include both the suspended sediment transport and bed load evolution and are based on the solutions of

- a three-dimensional advection-diffusion equation for suspended sediment transport for each active-layer sediment size class
- a two-dimensional mass-conservation equation for each active-layer bottom sediment size class
- a two-dimensional global mass-conservation equation for bed sediment covering all sediment size classes
- auxiliary relations which parameterize reference concentration, bedload transport, settling velocity, and vertical sediment diffusivity in terms of bed shear stress, vertical water diffusivity, and sediment properties

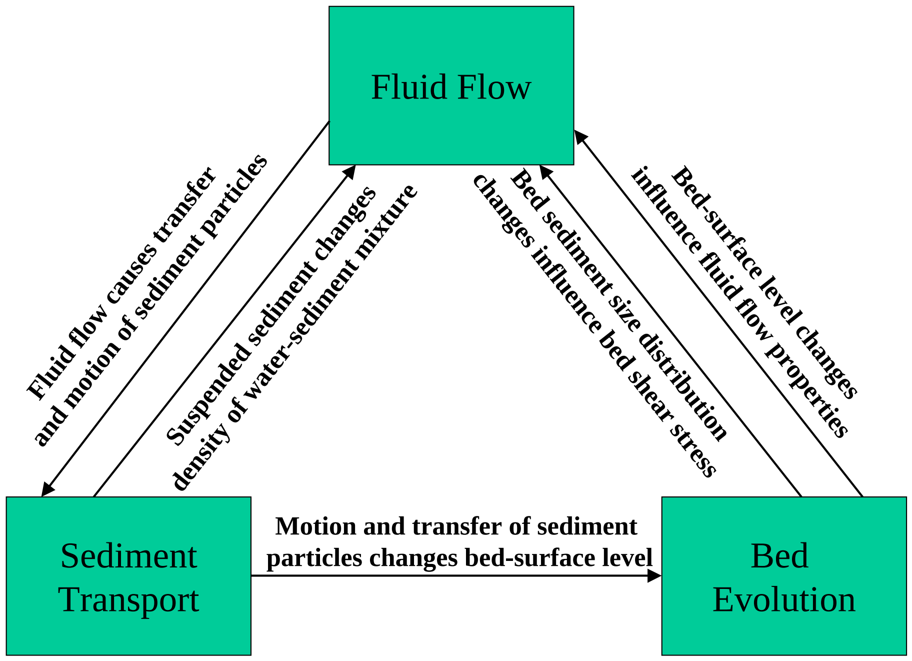
Fluid Flow
Fluid flow causes transfer and motion of sediment particles
Suspended sediment changes density of water-sediment mixture
Bed sediment size distribution changes influence bed shear stress
Bed-surface level changes influence fluid flow properties
Sediment Transport
Motion and transfer of sediment particles changes bed-surface level
Bed Evolution

Figure 2: The interaction between flow and sediment variables

Fluid flow, sediment transport, and bed evolution are parts of a coupled process. Fluid flow causes sediment transport by the motion, transfer, and deposition of sediment particles, thus resulting in bed evolution due to sediment transport. Similarly, changes in the bed-surface level causes changes in the flow field. The changes in the size distribution of the sediment particles at the bed surface changes the bed shear stress due to the changes in the roughness of the bed surface. Also, the density of the water-sediment mixture changes due to the suspended sediment in the fluid. The interaction between fluid flow, sediment transport, and bed evolution are illustrated in Figure 2.

The hydraulics model provides the sediment model with the flow field variables (velocities, depths, etc.). In turn, the sediment model provides the hydraulics model with changes in bed elevations (depths), distribution of active-layer size fractions (which governs friction coefficients) and changes in the density of the water-sediment mixture. The sediment computations fully communicate with the hydraulics computations at each time step, thereby providing a fully coupled system. The numerical solution of the suspended sediment equations are based on the QUICKEST scheme used in the computation of salinity and temperature in the hydraulics model. The numerical solution of these equations requires solutions of a system of nonlinear equations that are solved iteratively using the Newton-Raphson algorithm.

The flow computations are first performed until a “settling time” is reached, after which the sediment computations are activated. The sediment routines use the flow variables to compute the sediment concentrations and bed changes. First, the suspended sediment computations are performed and then the bed sediment computations are performed. During the suspended sediment

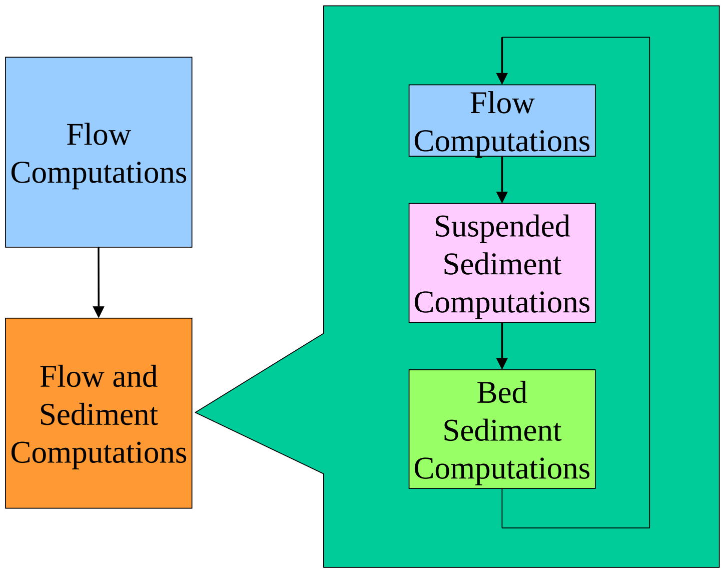
Flow Computations
Flow and Sediment Computations
Flow Computations
Suspended Sediment Computations
Bed Sediment Computations

Figure 3: Computation of flow and sediment variables

computation the deposition and erosion source terms are computed using the previous time-level concentrations at points in suspension above the bed. Using these new source terms the new time-level suspended sediment concentrations are computed. In the bed sediment computation the deposition and erosion source terms are computed using the new time-level concentrations in suspensions above the bed. Using these recomputed source terms the bed sediment computations are performed to yield new size-fraction distribution at the bed surface. The order of computation of the fluid and sediment variables is shown in Figure 3.

4 Strategy for Parallelization

Domain decomposition is used to distribute data among different processors. In order to minimize the idle time, static load-balancing is used to distribute the data such that each processor gets almost the same number of computational points. The partitioning and load-balancing is done in the pre-processing stage, wherein, separate grid files are generated for each processor along with other necessary information about the partitioning. Thus there is no need to allocate any extra storage or scatter the grid data when the parallel program is executed. Also at the end of the parallel computation each process writes the output into separate files and the post-processing routines combine these files and generate output files suitable for verification and visualization.

4.1 Pre-processing and Partitioning

To parallelize the computation, a three-dimensional spatial domain is decomposed into subdomains. Each processor is assigned a subdomain and is responsible for the computations in that subdomain. There are many ways to decompose a three-dimensional subdomain and there is no unique answer as to which decomposition strategy is the best. It depends, among other things, on the problems size, the machine parameters (communication speed vs. computation speed, for example), and the numerical algorithms used. The one-dimensional decomposition strategy is used here for parallelizing this code based on the following considerations:

- Straightforward implementation. In a one-dimensional decomposition, a processor needs to communicate with only two neighboring processors for coordinating the computations.

- The constraint to replicate sequential results (for verification of parallelization) requires minimal changes to the ADI algorithms used in the sequential code. The hydraulics computation in the external mode are based on the ADI algorithm in the I and J direction, and the computations in the internal mode as well as the sediment computations are based on the solution of tridiagonal systems of equations in the K direction. Decomposing in only one dimension requires no modification of solution algorithms of tridiagonal systems associated with the other two dimensions. The only modification needed is for the tridiagonal solver associated with the dimension being decomposed.

- For most grids used in the computations with dimensions $NX \times NY \times NZ$, the parameter NZ is usually much smaller than NX and NY. Therefore, it is more efficient to decompose in either NX or NY for parallel processing. In Fortran, typically NX corresponds to the first index and NY corresponds to the second index in a two-dimensional array. Decomposing in NY puts all boundary data in contiguous memory locations, which simplifies communications between subdomains.

- The parallel version of the wave model being used in the project coupling hydraulic, sediment, and wave models [5] uses a one-dimensional domain decomposition in the J direction. The implementation of a similar decomposition in CH3D-SED simplifies inter-model communication.

Based on the preceding reasons a 1-D decomposition in the J direction is used to partition the domain. If the grid size is $NX \times NY$, then a simple distribution along J direction on NP processors would be to divide NY equally among NP processors. However, the CH3D-SED code uses the I-blanking technique to facilitate the representation of complex geometry. Cells which correspond to dry land are not used in the computation; those cells that are used are referred to as computational cells or active cells. Hence dividing the NY columns equally among NP processors will not provide a uniform distribution in computational cells. For example, the Lake Michigan test case contains 67×138 cells in the horizontal plane with a total of 9313 cells. However, the grid utilizes only 3598 computational cells or active cells. Hence it is important to get a uniform distribution for the number of computational cells when the grid is partitioned for the parallel runs.

A simple load-balancing scheme is used during the partitioning phase. The complete grid is read and the number of computational cells along each I line (n_i) and the total number of computational cells in the grid (N) are computed. The approximate number of computational cells per processor equals the total number of computational cells divided by the number of processors ($\frac{\sum_{i=1}^{NY} n_i}{NP}$). The number of computational cells in each I line are added until the sum is less than the approximate number of computational cells per processor. Since the I line cannot be split between

No. of processors	With no load-balancing			With load-balancing		
	Index	Columns	Cell count	Index	Columns	Cell count
1	1-14	14	222	1-19	19	351
2	15-28	14	398	20-31	12	362
3	29-42	14	431	32-43	12	370
4	43-56	14	406	44-55	12	349
5	57-70	14	337	56-70	15	362
6	71-84	14	316	71-86	16	364
7	85-98	14	360	87-100	14	371
8	99-112	14	485	101-110	10	345
9	113-125	13	486	111-119	9	374
10	126-138	13	157	120-138	19	350

Figure 4: Sample distribution of computational cells with and without load-balancing for 10 processors. “Index” denotes the global coordinates and “columns” denotes the number of columns in J direction.

two processors, the distribution in the computational cells will vary slightly from one processor to another. A sample distribution for the Lake Michigan grid on 10 processors is shown in Figure 4. The first part of the table indicates the number of computational cells in each processor with the simple distribution of the columns while the second part shows the number of computational cells when the above mentioned load-balancing technique is used. It can be observed that in the former case the number of columns remains almost the same, and the number of computational cells varies. In the latter case, the number of columns varies and the number of computational cells per processor remains roughly the same. Without load-balancing the processor with fewer computational points must wait at each phase of computation to receive the boundary values from the neighboring processors with more computational points, which could lead to excess idle time and reduce the parallel efficiency.

Once the grid decomposition is determined an extra I row is added on both sides of each block to provide a single-cell overlapping region. Then the blocks with their overlapping regions are written out into separate files based on the processor number; and the distribution of the blocks in global coordinates is written to disk for use during the post-processing stage. Along with the grid files, other necessary files for tide, river, and wind data are also written to disk. The pre-processing and partitioning is performed sequentially on a single processor.

4.2 Parallel Computations

All of the required input files are generated during the pre-processing and partitioning phase. Each processor reads its corresponding input files and performs computation on the local domain. At the end of computation of each time step data is exchanged between the neighboring processors and computations for the next phase proceed in parallel in each processor. Other than the salinity and temperature computations, all computations in the hydraulics model use a single-cell overlap, hence 1-cell data is exchanged at the boundaries. The salinity and temperature computations as well as all of the sediment computations use a two-cell overlap, which requires exchange of two-cell data at the boundaries.

In the external mode, the vertically averaged velocities and the free surface displacement are computed using an implicit scheme which requires tridiagonal solves in the I and J directions. Since the data is decomposed along the J direction, the tridiagonal solves along the I direction do not require any data exchange, whereas, the tridiagonal solve along the J direction requires a parallel tridiagonal solver. Since the total time spent in this part of the program is less than 1% of the total time we have used a trivial parallel tridiagonal solver, in which each processor computes the coefficients of the system of linear equations that should be solved and the first processor proceeds with the forward solution phase and transmits boundary data to the next processor, the second processor starts the forward solution phase and passes the boundary values to the next processor. This process continues until the last processor. The last processor completes the forward solution phase and performs the backward solution and sends the boundary information back to the previous processor. This continues until the data reaches the first processor. After each processor sends the boundary values of the forward solve to the next processor, it has to wait until it receives the boundary values for the backward solution from the next processor. This idle time can create a bottleneck and degrade scalability when large number of processors are used. In order to reduce the idle time, one of the widely used techniques in the relevant literature is to overlap computation of the coefficients for the next J line with the idle time in waiting for the data from the backsolve for the current J line. Unfortunately, due to the use of the I-blanking technique an effective overlap of computation and communication requires extra book-keeping, check points, and makes the code very complicated. Hence it is not implemented in the current version of the code but left as a future enhancement.

Other than the external mode solution phase, the rest of the code uses implicit schemes in the vertical direction (K direction) only. Hence, there is no extra message-passing involved except for the exchange of boundaries for data blocks with overlapping regions. A flow chart for the parallel version of the application will look similar to Figure 1, except that there will be an exchange of boundaries after each phase of computation. These exchanges of boundary information are performed separately in a Fortran 90 module, thus, there are no message passing calls in the original code. Whenever data needs to be exchanged, the corresponding function call in the F90 module is called. The F90 module contains not only the message-passing routines but also all of the data associated with message passing (such as communicators, status and request objects, etc.), thus there is no need to include any information in the common block structure of the original code and therefore minimal changes to the original code are required. The use of F90 modules restricts all of the code changes to one module, thus making the code maintenance and enhancements easier.

4.3 Post-processing and Visualization

At the end of the parallel computation each processor writes the output variables into a separate file based on the processor number (rank). Using the grid distribution information generated during the pre-processing and partitioning stage, the separate output files are combined to form a single output file for each output variable and written to disk in a format consistent with the data structure used in the original sequential code. This enables the verification of the parallel code and utilization of existing post-processing tools for visualization. This process is performed sequentially on a single processor.

5 Test Cases

Code verification was performed using two test cases that were executed sequentially on the CEWES MSRC platforms. The results obtained from the sequential code were used as the baseline for comparison of the results obtained from the parallel version of the code. The following sections describe the rationale for using these test cases, input conditions used, output obtained, and the performance results for both test cases. The first test case (Lake Michigan) uses wind, temperature, and two sediment size classes for input, whereas, the second case (Old River) uses river, tide, and six sediment size classes.

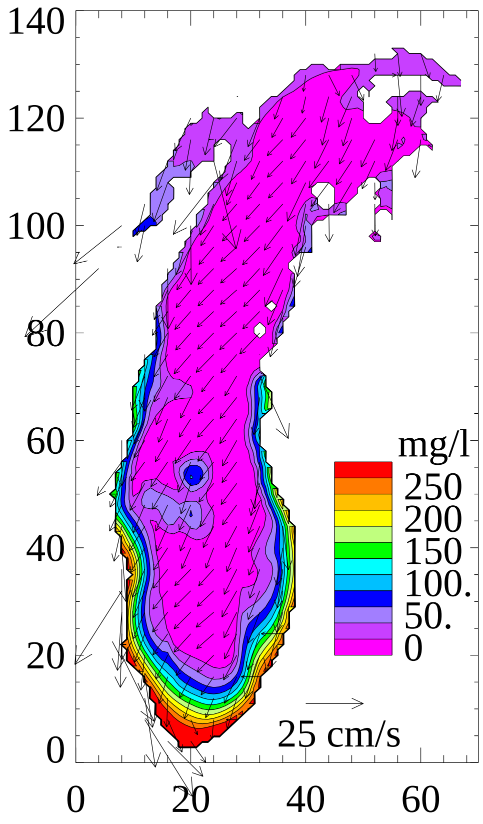

Figure 5: Sediment concentrations at 13 m depth and surface currents at hour-12 of the CH3D-SED Lake Michigan test case

5.1 Lake Michigan

A Lake Michigan test case has been chosen by Ohio State University (OSU) researchers to test the coupled hydraulics, sediment, and wave models since extensive field measurements and modeling programs are available. Also since the lake is a closed domain it is easier to implement boundary conditions correctly and more attention can be given to the coupling issues. The test case was setup to simulate the annual episode of spring sediment resuspension along the southern shores in Lake Michigan, a phenomenon that has been well demonstrated by satellite observations. The same test case is used here to test the coupled hydraulics and sediment models and is described below (see [6] for more detailed description of the initial conditions).

A 4-km grid with 67×138 points is used and the grid has 20 vertical layers with a denser distribution near the surface. The circulation and sediment transport are driven by a simplified wind field, which exists in the first day (linearly increases to 15 m/s during the first 18 hours, then remains unchanged until the end of the day) and becomes zero during the rest of the simulation period. Spatially, the wind is from the north and uniformly distributed over the lake. Two typical size classes of sediments (diameter 0.01mm and 0.1mm) are used in the simulation. The initial temperature is set to 4 degrees centigrade throughout the lake. Heat flux at the bottom and lateral boundaries and water surface are set to zero; this simplification is reasonable in a short simulation.

The above-mentioned initial conditions were setup for the current version of the code and tested on a single processor of the Origin 2000 and CRAY T3E by running 600 iterations with a time step of 2 minutes. The results obtained were used as the baseline for testing the parallel code. Figure 5 shows sediment concentration contours in mg/l at a depth of 13m, plus surface current vectors, for hour-12 of the simulation. Figure 6 shows a satellite image of sediment concentration at 1900 UTC on March 21st, 1998; the image is from the NOAA/NSF "Episodic Events - Great Lakes Environment" project [3]. Although the color bars in Figures 5 and 6 are different, it can be seen that the sediment plume predicted by the idealized CH3D-SED simulation is qualitatively similar to the observed plume.

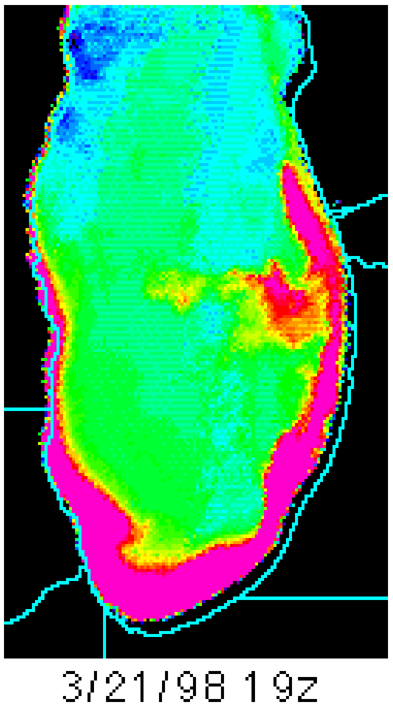

Figure 6: Observed sediment concentrations in southern Lake Michigan

The parallel version of the code was executed on the SGI Origin 2000 and the CRAY T3E platforms at the CEWES MSRC for various numbers of processors. Figure 7 provides the execution time on the two platforms for the Lake Michigan test case. These results were obtained by running the test case with the 69×138 grid for 600 iterations with a time step of 2 minutes.

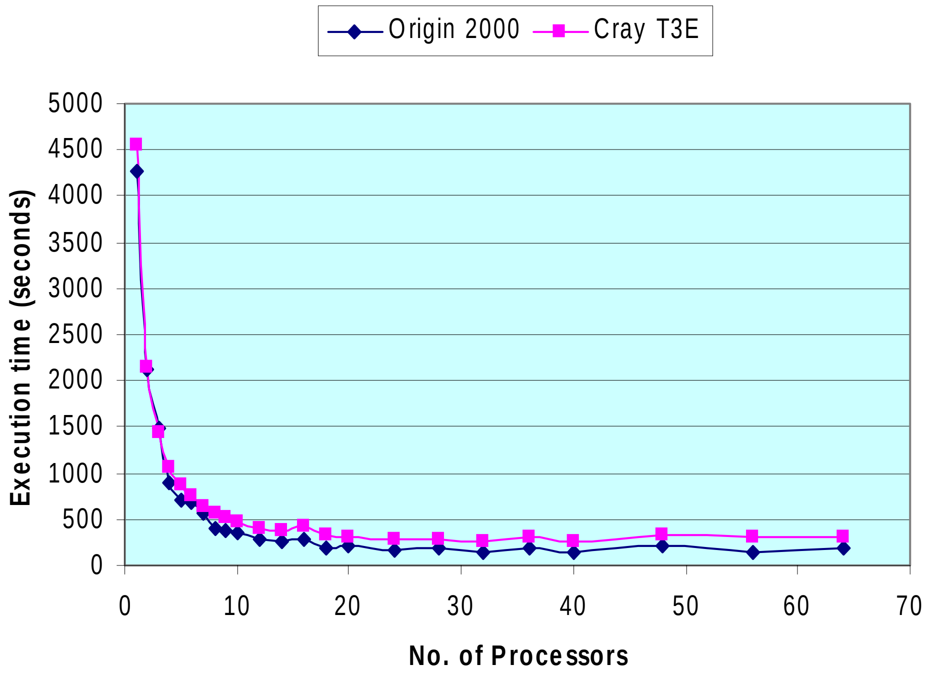

Figure 7: Execution time for the Lake Michigan test case on the Origin 2000 and CRAY T3E

It was observed that the execution time was reduced significantly up to 32 processors after which the reduction is not very significant. This behavior was expected since the number of I lines per processor will then be less than 5 and each processor has to exchange data after performing computations on less than 5 rows (less than 100 computational cells). With more than 32 processors the time spent in data exchange will be significant compared to the time spent in computation and thus the parallel efficiency goes down. Also when more than 32 processors are used, balancing the number of computational cells per processor will become a difficult task since we cannot split the I line between two processors and a single I line can cause significant load imbalance. The parallel tridiagonal solver used in the external mode adds additional overhead when large number of processors are used. We observe that when the number of processors are increased the execution time suddenly increases for a certain number of processors on both architectures. This is again due to the uneven distribution of the computational cells when a large number of processors are used. For a grid with more computational cells, the scalability of the parallel code would extend to a larger number of processors.

5.2 Old River

The Mississippi river at the Old River control structure complex was used in [4] as a prototype test case. The same test case is used here to test some of the features not demonstrated in the Lake Michigan test case. The key components that were tested in this case are the subroutines that model river and tidal boundary conditions. Six sediment size classes were used in this test case, which requires a significant amount of computations. A computational grid of size 50×345

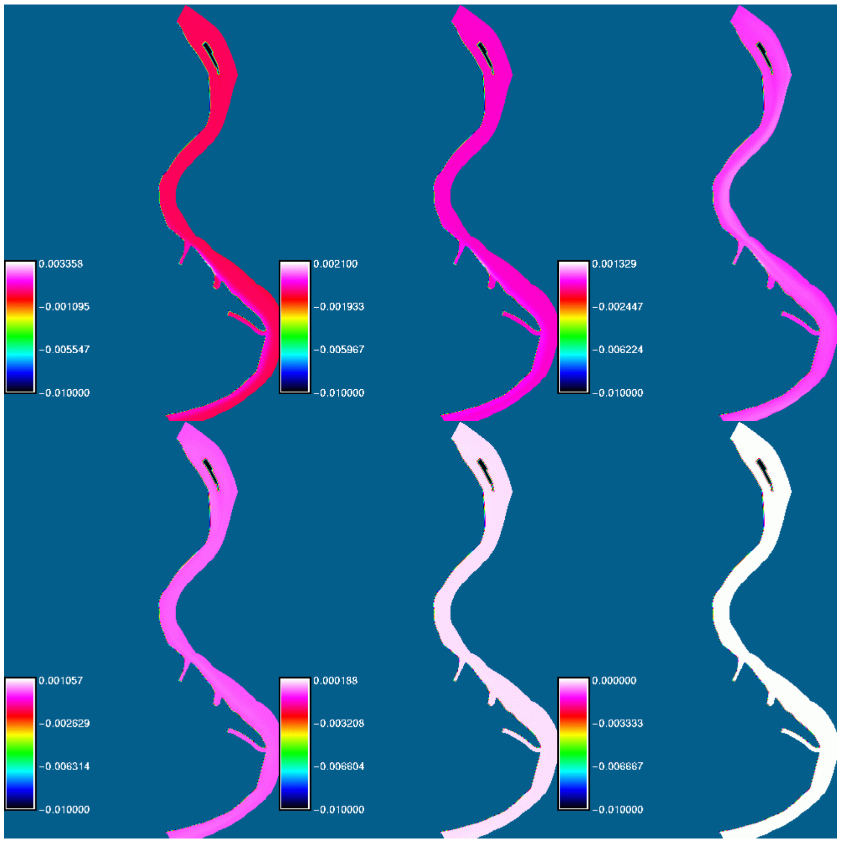

Figure 8: Output plots for the six different sediment size classes for the Old River test case after 8160 iterations with a time step of 15 seconds.

with 10 vertical layers is used and the simulation is performed for 200 iterations with a time step of 15 seconds. The sediment computations were switched on after 100 iterations. The simulation was performed on a single processor and then the results were compared with subsequent parallel results. Extensive performance measurements are not performed with this test case since the intention was only to test the functionality of the parallel version and make sure that identical results were obtained from the sequential and parallel codes. However, the parallel version of the code was used to perform a longer simulation on 32 processors of the CRAY T3E. The sediment concentrations for the six sediment classes used at the end of 8160 iterations is shown in Figure 8. In this case, the sediment computations were activated after 2400 iterations. The parallel version of the code has enabled DoD personnel to perform longer simulations in a shorter period of time.

6 Summary and Conclusions

The development of a coupled and scalable parallel hydraulics and sediment model was described in this report. The hydraulics model and the sediment model were first introduced and then the interaction between the two models was explained. Parallelization was carried out with a three-stage approach: pre-processing and partitioning, parallel computation, and post-processing. During the pre-processing and partitioning phase the grid is partitioned using a simple load-balancing technique to obtain roughly an equal number of computational cells per processor, with separate grid files generated for each processor along with the other necessary input files. All the processors performed the computation in parallel and exchanged data at the processor boundaries. At the end each processor wrote its corresponding part of the solution to disk. These individual files are later merged during the post-processing stage to generate output files suitable for plotting the results. Simulations were performed on two different test cases, Lake Michigan and Old River, to test the full functionality of the coupled model. The Lake Michigan test case was tested on the Origin 2000 and the CRAY T3E for various numbers of processors and the performance was analyzed. Significant reduction in the execution times was observed up to 32 processors, and with a larger grid, scalability would extend further. Extensive performance analysis was not performed on the Old River test case, but the parallel version of the code was tested for different numbers of processors and also used to perform a longer Old River simulation. The impact of this work at CEWES MSRC is to significantly reduce the time required for high-resolution sediment transport simulations by making use of the scalable nature of the high performance computing facilities. This work was also essential for the development of a scalable, coupled model of hydraulic, sediment, and wind-wave processes, which greatly affect military operations in the marine environment.

7 Future Work

It was observed that the scalability of the coupled hydraulics and sediment model tended to degrade when more than 32 processors were used. This is typical for simulations on parallel computers with a fixed problem size. In general, the problem size must also be increased with the number of processors in order to maintain scalability. Other reasons for this degradation were identified as the excessive time spent in message passing compared to the time spent in computation, the lack of good load-balancing, and the sequential nature of the parallel tridiagonal solver used in the external mode. One approach to improve scalability would be to consider a two-dimensional partitioning of the domain. The one-dimensional partitioning uses communication between two neighboring processors and the data exchange involves sending and receiving a complete I line. A -dimensional partitioning will require communication between four neighboring processors (three communications for those processors on the exterior of the processor grid) but would involve sending and receiving a piece of I line and a piece of J line. This would reduce the amount of data exchanged but increase the number of communications performed. To obtain better load-balancing with a two-dimensional partitioning we we have to investigate the use of more complicated partitioning schemes. Also with a two-dimensional partitioning the solution for the tridiagonal systems in the external mode will require an efficient parallel tridiagonal solver for both the I sweep and the J sweep. A two-dimensional partitioning provides an opportunity to explore the possibilities of using a scalable iterative method to solve the tridiagonal system instead of the direct method currently used. Furthermore, we could also investigate the possibility of using an approximate factorization scheme with an iterative method to solve the external mode instead of an ADI scheme. Thus the use of more sophisticated domain decomposition methods along with new numerical schemes could potentially improve the scalability and efficiency of the coupled hydraulics and sediment model.

References

[1] Raymond S. Chapman, Billy H. Johnson, and S. Rao Vemulakonda. User's Guide for the Sigma Stretched Version of CH3D-WES. Technical Report HL-96-21, U.S. Army Corps of Engineers, November 1996.

[2] B. P. Leonard. A Stable and Accurate Convective Modelling Procedure Based on Quadratic Upstream Interpolation. *Computer Methods in Applied Mechanics and Engineering*, 19:59–98, 1979.

[3] D. J. Schwab, D. Beletsky, and J. Lou. The 1998 coastal turbidity plume in Lake Michigan. Technical Report (in press), special issue of Estuarine, Coastal, and Shelf Science on Visualization in Coastal Marine Science, 1999.

[4] Miodrag Spasojevic and Jr. Forrest M. Holly. Three-Dimensional Numerical Simulation of Mobile-Bed Hydrodynamics. Technical Report HL-94-2, Iowa Institute of Hydraulic Research, August 1994.

[5] Miodrag Spasojevic and Jr. Forrest M. Holly. Three-Dimensional Numerical Simulation of Mobile-Bed Hydrodynamics. Technical Report 262, Iowa Institute of Hydraulic Research, September 1997.

[6] S. Zhang, D. Welsh, K. Bedford, P. Sadayappan, and S. O'Neil. Coupling of Circulation, Wave, and Sediment Models. Technical Report CEWES MSRC/PET TR/98-15, Ohio State University, 1998.

[7] J. Zhu, B. Johnson, P. Bangalore, D. Huddleston, and A. Skjellum. On the Parallelization of CH3D. Technical Report CEWES MSRC/PET TR/98-07, Mississippi State University, 1998.